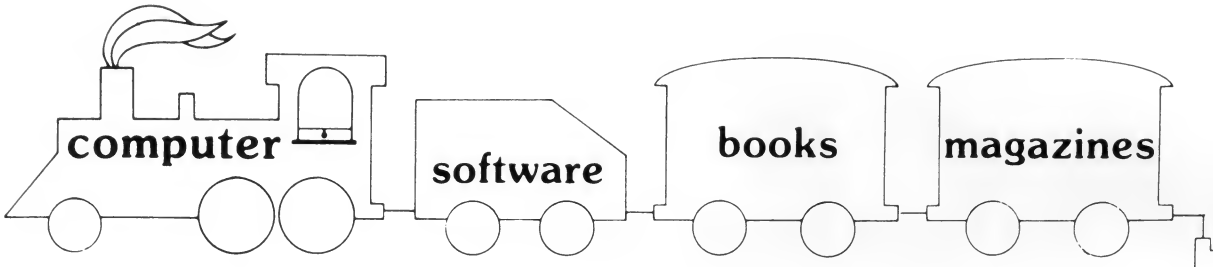
$2.25
March, 1983
Vol. 3 NO. 3

M.A.C.E. JOURNAL

"People For Computers"

* CRAIG CHAMBERLAIN'S "SHAPTAB"
* ATARI 1200XL USER EXPERIENCE
* ODESTA/ATKINS CHESS 7.0 REVIEW
* OSS MAC/65 & DATASOFT COMPILER



supplies

printers

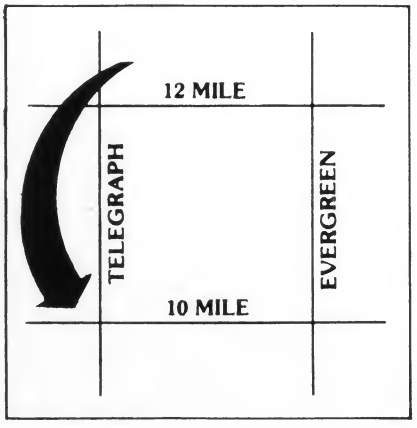
modems

monitors

classes

15 to 50% OFF ON ALL SOFTWARE IN STOCK

LIMIT ONE OF EACH TITLE
NO SALES TO DEALERS



micro station

24484 W. Ten Mile Rd. • Southfield, MI 48034
(½ block W. of Telegraph)
(313) 358-5820
Mon. thru Fri. 11a.m - 7p.m. Sat. Noon - 6 p.m.
*offer not valid with any other Discount

Sale runs now through March 31, 1983

ATARI 1200XL USER EXPERIENCE

by
M. Reichman and I. Chadwick

compiled and edited by
Greg Leslie - SYSOP
GREKELCOM BBS (405)-722-5056

The following is a series of messages from SIG*ATARI, written by Compuserve Atari SIG SYSOP Michael Reichmann and Ian Chadwick (both of "Mapping the Atari" fame).

#: 7983 Sec. 0
Sb: #1200 XL review
11-Feb-83 19:14:02
Fm: SYSOP*Michael Reichmann 70001,1003
To: All

I have been fortunate enough to have recieved a 1200XL for review. Over this weekend I will be posting from time to time, my ongoing impressions and comments. The evaluation that I'm doing is destined to be a magazine review, but I think our members might be interested in a blow-by-blow as I go along.

As far as I know this will be the first independent review of a 1200XL to appear anywhere! The 'review' in the latest ANTIC doesn't count, as it was written by an Atari Inc employee.

I should start off by saying that when the 1200XL was announced in December, I was one of the first to come down on it, as I was disappointed that it wasn't the wonderful dual-processor 16-bit machine that we've all been hoping for.

As I drove home yesterday afternoon, through rush hour traffic, with the 1200 in the front seat beside me, I decided that I really wanted to like it. It LOOKS 'sexy'. It LOOKS like the 'next generation' computer. Is it? Lets see.

Let me stress that this ongoing review is strictly my own opinions, and not those of this SIG, the other Sysops, or CIS. I also should say that all of these observations are being written as I go along, and I may contradict myself as I discover new things over the next couple of days. This machine is also a very early machine, and MAY NOT be representative of the units that will start appearing in stores in a couple of months.

First, the keyboard. It is one of the first things to grab you and it is a very good one. The feel is perfect for a touch typist, and with the (Atari) key now moved and the left SHIFT key enlarged, there should be very few complaints about it. What you will simultaneously discover, is that there is NO BUILT IN SPEAKER. The key clicks are now produced through the speaker, as are the buzzer and the output of ? CHR$(253).

Is this a good thing? Hummm. It means that you won't have to put in a switch to turn the speaker off, so that typing in the middle of the night won't wake your spouse, (the way I have), but it also will create problems for users who keep their TV sound off or particularly, those that use monitors. Most monitors don't have audio, and many software producers count on the console buzzer, to be able to give the user an auditory warning.

I won't go into a detailed physical description, as you've all seen photos by now and also read the nuts-and-bolts descriptions. The machine is pretty, and though I thought the chromed special function key strip, along the top of the keyboard, is shiny, it does not glare. What does glare, is the plastic cover above that, that covers the key labels and function lights. Not a great annoyance, but unnecessary.

As you know, START, OPTION and SELECT have been moved from the right hand side to the top row. This is the first major problem that I discovered. Many programs now make heavy use of these keys, Text Wizard, for example, and with them on the right, you can use the left hand pinky to hold one down and the right hand for an Alpha key. In the short time I've used it, I find the 1200's placement along the top very awkward, and if the alpha key you also need to press is on the right side of the keyboard, your hands end up crossing.

It's interesting to note that Commodore copied the Atari's when they brought out the Vic 20 and then again the C64, by putting their special function keys in exactly the same place, and even looking the same. In my opinion, Atari got it right the first time, and should have left well-enough alone.

Cartridges are a snug fit. The slot is on the left side. I have just heard that some non-Atari

carts will not fit the 1200. I don't have any handy and therefore can not immediately verify this myself, but a very reliable source told me last night that Synapse carts, at least, won't physically fit.

Reports have had it that the color circuits are improved. On the unit I have, the colors are VERY bright, but it looks to me like all they've done is boost the chroma levels. On some images the colors are so saturated that double imaging appears. I've fiddled with my TV, which produces a very fine picture on cable input, and from my 800, and can't find a setting that produces less saturated colors, without the color fringing.

The monitor output is absolutely dreadful. It may just be that this unit is faulty, (why give a reviewer a faulty machine?), but at first I thought that the video was kaput. After turning up the black level and contrast quite a bit, I was able to get a picture, but it's very weak and undersaturated; the opposite of the RF output. It is so fuzzy, and unreadable on text, that if I had bought this machine, I would return it for an exchange. If it turns out that this flaw is not unique to this particular machine, but endemic to all 1200's, I can state that no one with a monitor will be able to use one.

Self tests: On power-up, with no cart, you see a pretty Atari logo with scrolling multi colors. Cute. This though, replaces the memo-pad mode, which I must say I find VERY handy. My son is now almost 6, but from the day I brought home my 800 a couple of years ago, he has enjoyed simply typing and seeing the letters appear on the screen. He then discovered the CONTROL character set, and loves creating pictures. This is now gone. You can't really do it in BASIC since the screen fills up with 'Syntax Error' messages. Not a big deal, but I'd miss it.

The HELP key, when pressed without a language installed, runs a series of built in self-diagnostics, including a test of ROM, RAM, Audio, Video and keyboard. I have run the full set of tests and the machine tests out perfectly. I've rerun the ROM/RAM test several times. (I'll tell you why in a minute).

The tests are nice to have, but I wonder how much room in the new 14K OS they take up. The AV test for example displays a music staff and plays and displays notes, like the Music Composer cart. Hummm. I wonder how soon people will tire of confirming that their machines are in fact OK?

I've spent some time on the fact of the built-in diagnostics, and that my evaluation unit test out perfectly, because I'm about to let you know that their is a ***MAJOR*** problem. If the diagnostics weren't there, I would not have said anything, and just returned the unit for another one, assuming that the new ROMs were bad. Well, The tests say that they are good, and being the dumb consumer that I am, I will state the following: A GREAT NUMBER OF EXISTING PROGRAMS FOR THE 400/800 WILL NOT RUN ON THE ATARI 1200XL.

In the past couple of hours, I have determined that the following disks will not boot: Filemanager 800+, Textwizard, Shamus, Protector II, and Slime.

I plan to run through my entire software library and report on any others that I find won't run.

Now listen! I may be wrong. I may be doing something wrong. What I am saying though, is that I'm acting like a naive consumer. I've read the owners manual, and I've booted the disks. They just hang, each at various stages of booting. I think there IS an OS conflict, probably with both copy protection routines and some 'illegal' entry points, but why hasn't Atari said anything about this before??

Apple has at least publicly stated that the new IIe is NOT 100% downwardly compatible. Is Atari not able to extend the same courtesy to its customers?

I hope that I'm wrong!! I really do. Until I've been able to determine exactly whats going on here, I don't want to say much more about this but if anyone from Atari reads this, please reply and tell me it isn't so. (Or at least confirm the sad truth.)

We have now taken the 1200 apart. It is indead a single board computer and my, is it neat. It opens very simply, and looks to be well constructed, and also VERY inexpensive to manufacture. The whole board is about as thick as an issue of Compute!, and other than for the keyboard, that's it.

The rumours about the board being potted, (sealed), are untrue. All of the chips are freely

accessible. I'm not a technician, so I can't say much more about it, but I did not see ANY way to plug anything into anything. It is indeed an appliance computer. What you see is what you get.

I have done a dump of the OS and the elves will be madly at work for the next week doing a disassembly and analysis.

Ian Chadwick, author of Mapping The Atari, has taken it for the rest of the weekend, and he will be doing further testing and evaluation.

What do I think? All I'll say at this point, till I've had a chance to digest everything noted thus far, and benefit from Ian's observations, is that I'M VERY GLAD to be using my 800 again, and if a 1200 were GIVEN to me, I doubt if I'd use it! Who knows; Atari probably will sell a raft load of them. Just not on my recommendation though.

Well weve examined the dump of the OS, and here are a few preliminary observations.

Pages two to four have some major changes. The interrupt vectors starting at decimal 512 now seem to point to the previously unused 4K ROM block starting at $C000. This probably explains why certain programs such as Text Wizard and some games won't run on the 1200.

The spare locations in this area are also now used, so any programs that use them will stumble.

What I now want to know is this.... OK Atari, some software that makes illegal entries won't run on the 1200. The same thing happens with the Apple IIe, and I suppose software authors should know better. BUT, when Apple announced the IIe they told the world that this was the case, and more to the point, 6 months before introduction they were working with the major software houses, so that when the IIe hit the stores, there was compatible software.

As for telling us about compatibility; well, I for one haven't heard a word. This then is an open question to Atari Inc. HAVE YOU BEEN WORKING WITH THE SOFTWARE HOUSES TOWARDS PRODUCING MODIFIED VERSIONS OF THEIR PRODUCTS THAT WILL BE COMPATIBLE WITH THE 1200XL?

A couple of other notes. On a 400/800, typing GR.12,13, or 14 gives an error 145, "Bad screen mode handler". On the 1200 these modes now work, and are the missing Antic modes such as the famous mode 7.5 used by Micropainter.

Unfortunatly, the user manual says nothing whatever about any of this so we're fishing around in the dark.

One last thing for now. The color clock is one pixel different between a CTIA and GTIA; on the 1200 it seems to be one more pixel again, so that artifacting that was, say, RED on an 800 with CTIA, and GREEN on an 800 with GTIA, now seems to be RED again. Or does it? Anyhow, the artifacting appears to be the same as on a CTIA. A minor point. On a 400/800 typing BYE put you into memo pad mode and now it puts you into the Atari logo and HELP mode. On a 400/800, if you hit RESET though, it would put you back into your BASIC program. On the 1200XL, RESET from BYE will do a cold start reboot, so watch out.

POSTSCRIPT:
From IAN CHADWICK 70001,1002:

THERE IS MORE WRONG WITH THE 1200 THAN MICHAEL HAS STATED HERE. The interrupt vectors, stored in RAM from 512 on, have all been redirected to a new area of memory; that 4K block at $C000 now used for the OS. Any programs which make direct jumps instead of through the RAM vectors, will hang. Also, in a benchmark BASIC test, Atari BASIC proved about one half the speed of the same BASIC with the Newell FASTCHIP installed on the OS board. As expected. Quite a number of vectors seem to have been changed and some previously unused bytes in the RAM area pages 0 to 5 are now being used. Hope Atari brings out tech manuals & OS listings VERY soon!

One useless kludge they made: rather than free the joystick/paddle registers that are now useless since there are only two ports, they merely had the unused registers mirror the values in the used pair. Sigh. Kludge kludge kludge. I also noted that there was more difficulty getting AUTORUN.SYS files to load than others. This may be due to OS changes as well. The colour gain is so high that with a SETCOLOR value for luminance at 14, my tv screen went crazy. Too much signal. Let's hope the Atari folks hear what we're saying.

SHAPTAB

by

Craig Chamberlain

[NOTICE: The SHAPTAB project will appear in a national publication soon. Therefore, contrary to normal MACE policy, the SHAPTAB article and programs may not be reprinted or distributed beyond MACE members. The article and programs are protected by copyright.]

A quick glance at current literature about ATARI graphics would seem to indicate that players, missiles, and redefinable character sets are getting all the attention. What about good, old fashioned bit mapped graphics? Does anybody realize that of the big three consumer computers (APPLE, ATARI, COMMODORE 64), bit mapped graphics on the ATARI is the most versatile and easiest to use?

There are a number of reasons that make ATARI bit mapped graphics superior. Let's look at what the other guys have to offer:

APPLE][

This is simple. The APPLE has only two bit mapping modes. The lo res mode supports sixteen colors and is very nice, but the resolution is only 40 by 48 (hence, "lo res"). The hi res mode, according to APPLE, supports eight colors. They are orange, green, blue, purple, white, white, black, and black. Yes, eight colors. I count four colors, plus white and the background. But there's another catch. Some color combinations for contiguous pixels are not possible. Are you confused? Well, don't get confused yet! We still have to talk about the...

COMMODORE 64

This is not so simple. The graphics on the C64 are weird! This computer has a standard bit map mode and a multi-color bit map mode. The mapping for both of these modes is set up just the same as for characters modes! The first eight bytes build a column of the display, for the first row. The ninth byte starts the second column, and so on. The display data for the standard mode (320 by 200) is separate from the color data. The multi-color mode (160 by 200) is a tangled mess of color registers, video matrix high and low nibbles, and four-bit color codes. Now is when you should be confused!

ATARI

Enough talk about the other guys. Really, the above computers are very good machines, and their graphics are often quite satisfactory. The COMMODORE 64 does allow more colors on the screen than the others. There are, however, several reasons to justify the statement that ATARI bit mapped (called "playfield" in ATARI parlance) graphics are superior. Here are my reasons:

1) The ATARI supports several modes of varying resolution and memory requirements. In fact, it has eleven distinct bit mapped modes. They are low, low-medium, medium, and high resolution in two and four colors, a very high resolution mode, and specialty modes in medium-high resolution (80 by 192) for sixteen colors (eight times better resolution than APPLE), sixteen shades (nobody else has that), and nine of sixteen colors, each with independent luminances (nobody else has that, either). Such a wide variety means that ATARI graphics are more flexible and better suited for handling individual applications.

2) The ATARI bit mapped modes have a simple, easy to understand memory layout, one that corresponds with the manner in which the points are displayed when plotting.

3) There are no placement restrictions whatsoever. You can put any pixel next to any pixel, each pixel contains image and color information, and that is that.

4) ATARI graphics modes support color indirection. The APPLE has no such thing (you are stuck with the exciting orange, green, blue, and purple). The COMMODORE 64 kind of supports color indirection, but is still limited to a palette of sixteen colors (no luminance selections) instead of one hundred twenty eight on the ATARI.

Those are some of the reasons why I think ATARI overall has the better playfield graphics. Unfortunately, nobody seems to appreciate this fact, being more interested in

player/missile and character set graphics. Why is that?

PLAYER/MISSILE GRAPHICS

This graphics function (known as "sprite graphics" on the COMMODORE 64) features low memory consumption and, more importantly, easy movement and animation of displayed objects. Doing this animation in playfield graphics is difficult because the background display must be continually erased and restored. It is not usually possible to erase and replot that many points fast enough using playfield graphics.

CHARACTER SET GRAPHICS

With this method of graphics, one character defines a whole eight by eight image. The entire image can be displayed with just one memory access. Animation is just as easy as plotting a different character on top of the first one. To display or change the image in playfield graphics would require several points to be plotted, one for each point in the image. Again, this would be too slow and tedious.

These graphics functions were created to fill a need. The "problem" that these methods circumvent is the difficulty in quickly plotting large numbers of pixels in bit mapped graphics.

What if there was a way to quickly plot lots of points on a bit mapped screen? Would that render player/missile graphics and redefinable character sets obsolete? No - these graphics features have definite uses and advantages. But would fast plotting enable one to do things that can't be done with player/missile graphics or redefined character sets? YES! Now do I have your interest? What if I then told you that you could use this fast plotting in your very own programs? For sure, that must have your interest!

SHAPTAB

This is an ATARI version of a graphics utility commonly known as a shape table. SHAPTAB supports all of the normal operating system bit mapped modes (3 to 11) and, as will be shown later, can even be useful in text modes (0 to 2). SHAPTAB has the following advantages over players, missiles, and character sets:

1) It is now easy to plot the same (or different) objects at various points on the screen (one quickly learns that four players are not enough).

2) Shapes are not limited to eight by eight matrices. A shape generated by SHAPTAB can be any pattern in any size.

3) There are no positioning restrictions. In terms of where it is possible to position a character, text modes have low resolution. The horizontal resolution is usually either twenty or forty, while vertically it is twenty four or twelve, even though the actual resolution of individual points may be much greater.

When combined with the ATARI's capabilities of color indirection, SHAPTAB becomes a very powerful little tool that opens up a whole new world of playfield graphics to many new applications. Best of all, SHAPTAB can easily be combined with your own programs written in ATARI BASIC.

WHAT IS A SHAPE TABLE?

A shape table is a sequence of instructions to move the graphics cursor and plot points. Movement is usually restricted to one position away from the previous point, so the position of one point is relative to the position of the previous point. For example, the shape table to draw a square would consist of eight entries, like this: up, up, right, right, down, down, left, left. Normally four or eight directions are allowed for positioning. At each position, a point is plotted, in one of the colors supported by the graphics mode. Each instruction of the shape table includes the direction and color information. A shape drawn with a shape table appears as a chain of dots which twist and turn and change color to create an image. It would be just as if a little man went stumbling about your screen, dropping splashes of various colored paints along the way.

FEATURES OF SHAPTAB

The current implementation supports eight directions and sixteen colors, so it can accomodate any shape in any graphics mode. There is a special instruction that indicates

the end of the shape table. Each entry in the shape table is one byte, so SHAPTAB is a much more efficient method of representing images than by maintaining cursor coordinates and color values. Certainly the most useful feature of SHAPTAB is the built in extra level of color indirection. It is possible to define a shape in one playfield but then change that playfield designation at the time when the shape is drawn in the application program. One shape definition can be used to draw the same image in various different playfields, all on the same screen. This indirection allows SHAPTAB to be useful even in text modes.

HOW SHAPTAB WORKS

For fast plotting, SHAPTAB bypasses the CIO utility of the operating system and directly calls the display handler. Specifically, it calls S:PUT. SHAPTAB could be made to plot even faster if it had its own plotting routine designed for a specific screen mode, but this would not suit our general purposes.

HOW TO USE SHAPTAB

The SHAPTAB function is accessed by the use of a USR (user defined) function in ATARI BASIC. The normal syntax for this statement with SHAPTAB would be something like U=USR(SHAPTAB,SHAPE) where SHAPTAB is the address of the machine code SHAPTAB routine, and SHAPE is the address of the start of the shape table to be used. For example, one might store the SHAPTAB routine in a string, say S$, so SHAPTAB=ADR(S$). If the shape table were also stored in a string called SHAPE$, it would be a simple matter to let SHAPE=ADR(SHAPE$).

When the USR function is called, the shape will be plotted, starting at the current cursor position, and the status code of the operation will be returned in the variable U, or whatever variable is used. If U had a value greater than 127 then an error occurred. For example, U having a value of 141 indicates that the cursor went out of range. The cursor position before the SHAPTAB call could be set by a POSITION statement, or if PLOT or DRAWTO have been used, the most recent point plotted will be the current cursor position.

As an added convenience, there is another syntax form for the USR call that lets the position be specified before the shape is drawn. U=USR(SHAPTAB,X,Y,SHAPE) will do this for you, provided X and Y are legal cartesian coordinates. This is the same as saying POSITION X,Y : U=USR(SHAPTAB,SHAPE). It's just more convenient.

If the second form is used, be sure that the X and Y values go before the SHAPE. The address SHAPTAB must always come first, and SHAPE must always be last. The SHAPTAB routine will detect an illegal number of parameters in the USR call, and in that case it will perform a system reset. SHAPTAB cannot, though, detect the case of having the correct number of parameters in the wrong order. Such a mistake could cause a system crash. You will have no problems as long as you follow the syntax forms given above.

VICTORY FOR BILL WILKINSON

In response to merry threats of bodily mayhem from Bill Wilkinson, SHAPTAB does not use any of page six. None at all! The main routine for SHAPTAB is one hundred and two bytes long, and is relocateable, so it can be placed anywhere in memory, and probably in a string. The routine does require some data in a specific place, however, so a small portion of the second half of page four is used. But not page six! Are you happy, Bill?

I should also point out that SHAPTAB uses page zero locations $D0 and $D1.

BLUE LIGHT SPECIAL!
EXTRA LEVEL OF INDIRECTION!

Oh yes. Here is how that playfield mapping works. There is a table sixteen bytes long starting at location 1172. The first byte is zero, the next byte is one, and so on up to fifteen. The sixteen color values from the shape table map directly onto this table. But if the second byte in the table, which normally has the value one, were changed into something else, such as a three, then any shape created using playfield one will instead be drawn in playfield three. What does that gain you? Well, your one shape definition in playfield one can now be used to plot the same shape several times, once in playfield one, once in playfield two, etc. simply by changing

RITE WAY'S

DISCOUNT PRICE!

EVERYTHING DISCOUNTED EVERYDAY

HIGH PERFORMANCE FLOPPY DISK **$1.79**

CASH REGISTERS 21 DEPT. **$495.**

THE DISCOUNT Computer SOURCE

OPEN
10 a.m.-9 p.m.
Mon.-Sat.

DISK FILE HOLDS 60 DISKS $19.95

SOFTWARE SOFTWARE

PRINTERS **EPSON**

MX-80 **$469.**

PAPER $15.95 CASE 1500 SHEETS

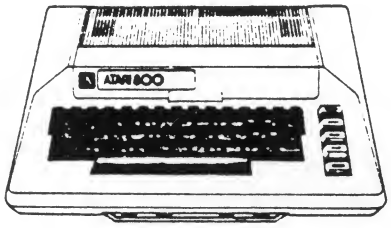
ATARI

Software For...
ATARI
APPLE
VIC 20
TI 994A
SINCLAIR

OVER 2000 PROGRAMS IN STOCK

Pro/Writer

$479.



ATARI 800 with
48K $499

What do we say about competition? We welcome it!!!

16K RAM

USED	NEW
29.95	49.95

Computers AND people

48k FOR 400
$99.

RITE WAY ENTERPRISES

8262-12 MILE RD.
WARREN MI. 48093
313-751-2454

22027 MICHIGAN AVE.
DEARBORN MI. 48124
313-562-3178

that one byte in the table. Or, you could rotate the bytes around to get a different effect. Another use of this feature is for erasing. Set all sixteen bytes to zero. Then any shape that was previously drawn can be erased just by drawing on top of itself. This indirection table is really quite handy.

SHAPTAB USED WITH CHARACTERS

The values in the sixteen byte table just described are not limited to only values zero through fifteen. The values can be in the range from zero to two hundred fifty five. This gives full access to a character set. Perhaps you are using character set graphics and have an image so big that it takes four characters to hold it. Rather than using four PLOT statements, set the first four bytes in the table to the proper character codes, and use SHAPTAB! The shape definition in this case would still be in playfields zero through three, but the table does the conversion to the character codes.

POSSIBLE APPLICATIONS

The SHAPTAB function does have many uses. It can do many of the things normally done with characters, without the trouble of redefining a character set. SHAPTAB could even be used to put lettering on a bit mapped mode, so you have text right with your hi resolution pictures. Although it was not intended to be used for this purpose, SHAPTAB can provide a cheap form of animation. To do this, it is necessary to include points drawn in the background color as part of the shape definition, so that the shape erases part of itself as you move it. Or, for animation without movement across the screen, you could alternate shapes and keep drawing at the same place on the screen, for another kind of animation effect. Any time you need the same shape at different places on the screen, SHAPTAB can be of use. There are many other uses of SHAPTAB, some of which will only become evident after experimentation.

SUMMARY

SHAPTAB is a great little graphics tool that everybody can put to good use. It is easy to use, has many applications, and breathes new life into the sadly neglected playfield graphics of the ATARI.

ACKNOWLEGMENTS

The following persons have participated in the SHAPTAB project:

Jeff Abel
Glen Carbone
Craig Chamberlain
Harry Morris

SHAPTAB was originally created by the request of Harry Morris.

SHAPTAB is part of a series of graphics extensions for the Atari to be published nationally. All criticisms of the SHAPTAB programs and documentation, including points regarding clarity and accuracy, or other comments, would be appreciated. What we especially need are demonstration programs showing how you have put SHAPTAB to use.

PLEASE! Send your criticisms, comments, and demonstration programs to:

ALLIANCE SOFTWARE
17094 DUNBLAINE
BIRMINGHAM, MI 48009

SHAPTAB

by
Craig Chamberlain

```
29000 ? CHR$(125):? "SHAPTAB":? "by Craig
Chamberlain 2/22/83":?
29010 RESTORE 29030:FOR K=1152 TO
1171:READ P:POKE K,P:NEXT K:FOR K=0 TO
15:POKE 1172+K,K:NEXT K
29020 DIM S$(102):FOR K=1 TO 102:READ
P:S$(K)=CHR$(P):NEXT
K:SHAPTAB=ADR(S$):RETURN
29030 DATA 173,23,228,72,173,22,228,72,
152,96,255,255,0,1,1,1,0,255,255,255
29040 DATA 104,201,1,240,17,201,3,240,
3,76,116,228,104,133,86,104,133,85,
104,104,133,84,104,133,209,104,133,208
29050 DATA 160,0,177,208,48,
50,230,208,208,2,230,209,72,74,74,74,
74,170,189,138,4,24,101,84,133,84,
189,140,4,170
29060 DATA 24,101,85,133,85,138,48,1,
152,101,86,133,86,104,41,15,170,188,
148,4,32,128,4,152,16,7,132,212,169,0
29070 DATA 133,213,96,162,2,181,90,149,
84,202,16,249,208,182
29099 REM SHAPTAB by Craig Chamberlain
2/22/83 COPYRIGHT 1983 ALLIANCE
SOFTWARE
```

SHAPTAB DEMO

USING GRAPHICS 9
by
Craig Chamberlain

Merge the SHAPTAB program with this listing before running.

```
100 GOSUB 29000:RESTORE 200:READ J:DIM
SHAPE$(J)
110 FOR K=1 TO J:READ
P:SHAPE$(K)=CHR$(P):NEXT
K:SHAPE=ADR(SHAPE$)
120 GRAPHICS 9:FOR Y=160 TO 30 STEP
-1:P=USR(SHAPTAB,10,Y,SHAPE):IF P>127
THEN STOP
130 NEXT Y:FOR Y=30 TO
160:P=USR(SHAPTAB,45,Y,SHAPE):IF P>127
THEN STOP
140 NEXT Y
150 GOTO 150
190 END
200 DATA 59,1,18,19,20,21,22,23,24,
25,26,27,28,29,30,31,62,61,60,59,58,
57,56,55,54,53,52,51,50,49,79,94,93,
92,91
202 DATA 90,89,88,87,86,85,84,83,82,
81,114,115,116,117,118,119,120,121,
122,123,124,125,126,127,255
999 REM DEMG9  by Craig Chamberlain
2/23/83
```

HOW TO USE THE SHAPTAB PROGRAM

by
Craig Chamberlain

The SHAPTAB program is the one that you combine with your own program in order to do the fast shape table plotting. It is in the form of a subroutine which you call once as part of the initialization in your own program. This installs the SHAPTAB routine in memory so that you can then do the USR call. You will have to write your own code to load in the shapes and store them in memory.

To merge the SHAPTAB function with your program, type in the following listing and save it to disk or cassette by using the LIST command. Do not use SAVE or CSAVE. Next, LOAD (or CLOAD) your own program and then use ENTER to add the SHAPTAB lines to your own program. In the initialization section of your program, which should be executed only once per each RUN, place the statement GOSUB 29000. After that point the variable SHAPTAB will be defined for use in USR calls. Please note that the SHAPTAB routine is stored in a string called S$, but this could easily be changed with no problems.

At this point, the SHAPTAB function has been installed, but it has no shape tables on which to operate. There are two methods to retrieve defined shapes. One is to use OPEN and GET, the other is to simply use READ and DATA.

The first method is preferable because it takes less memory and is faster. Unfortunately, it is not as easy to do this on a cassette based system as it is on one with a disk. The format of the save files from SHAPEDIT is two bytes, in lo and hi order, that tell how many bytes follow to define the shape. A typical code sequence might go like this:

```
OPEN #1,4,0,"D:SHAPNAME.SHP"
GET#1,LO : GET #1,HI
SHAPELENGTH = 256*HI + LO
DIM SHAPE$(SHAPELENGTH)
FOR K = 1 TO SHAPELENGTH
 GET #1,P
 SHAPE$(K)=CHR$(P)
NEXT K
SHAPE = ADR(SHAPE$)
```

The second method is convenient for cassette systems, but will work just as well on disk systems. We will introduce the standard that the first element in the DATA defines the length of the shape table. Here is an example of a possible code sequence:

```
RESTORE linenumber
READ SHAPELENGTH
DIM SHAPE$(SHAPELENGTH)
FOR K = 1 TO SHAPELENGTH
 READ P
 SHAPE$(K)=CHR$(P)
NEXT K
SHAPE = ADR(SHAPE$)
```

After either of these code sequences has been executed, the string SHAPE$ will be defined and the variable SHAPE will point to the beginning address of the table. You are now all set for a USR call. Remember that the syntax for such a call is:

```
U=USR(SHAPTAB,SHAPE)
        or
U=USR(SHAPTAB,X,Y,SHAPE)
```

By using several strings and variables such as SHAPE1, SHAPE2, and so on, it is possible to have many shapes defined simultaneously.

So now the only question should be, for cassette owners, how do you translate the shape stored as a cassette file by SHAPEDIT into a bunch of DATA statements? Well, we have one last program here to do just that. Program DATAMAKR builds DATA lines which can be merged with your program using ENTER in the same way as before. The operation of the program should be self explanatory.

I should mention one other thing here. The number at the beginning of a save file is one larger than the number of entries displayed by SHAPEDIT for that shape. That is because the save file includes the shape table terminator, which happens to be a 255.

Remember that in your program, besides doing the setup for SHAPTAB and SHAPE, you may also want to put certain values in the indirection table, or do any necessary SETCOLOR statements.

This concludes the SHAPTAB project. I sincerely hope you will give SHAPTAB a try, and give ATARI playfield graphics a chance to show their stuff!

DATAMAKER

by

Craig Chamberlain

```
100 DIM S$(120):IF PEEK(87) THEN GRAPHICS
0
110 ? CHR$(125):? "DATA STATEMENT
GENERATOR"
120 ? "by Craig Chamberlain  8/15/82":?
:TRAP 300
130 ? "Reading cassette file...":OPEN
#1,4,0,"C:":?
140 GET #1,LO:GET
#1,HI:SIZE=256*HI+LO:DIM BUFF$(SIZE):FOR
K=1 TO SIZE:GET
#1,P:BUFF$(K)=CHR$(P):NEXT K
145 CLOSE #1
150 ? "STARTING LINE NUMBER";:INPUT
C:IF C>32767 OR C<>ABS(INT(C)) THEN 150
160 ? "LINE NUMBER STEP";:INPUT D:IF D=0
OR D<>ABS(INT(D)) THEN 160
170 ? :? "Writing cassette file...":OPEN
#1,8,0,"C:":?
180 S$=STR$(C):S$(LEN(S$)+1)=" D.":
S$(LEN(S$)+1)=STR$(SIZE):K=0:GOTO 210
200 S$=STR$(C):S$(LEN(S$)+1)="
D.":S$(LEN(S$)+1)=STR$(ASC(BUFF$(K))):
IF K=SIZE THEN 220
210 K=K+1:S$(LEN(S$)+1)=",":
S$(LEN(S$)+1)=STR$(ASC(BUFF$(K))):
IF K<>SIZE AND LEN(S$)<107 THEN 210
220 ? #1;S$:C=C+D:IF C>32767 THEN ? "LINE
NUMBER OUT OF RANGE":GOTO 240
230 IF K<>SIZE THEN K=K+1:GOTO 200
240 CLOSE #1:END
300 ? "I/O ERROR ";PEEK(195):CLOSE #1
999 REM DATAMAKR by Craig Chamberlain
8/15/82 revised 2/24/83
```

SHAPTAB DEMO 2

"3D" GRAPHICS

by

Harry Morris

Merge the SHAPTAB program with this listing before running.

```
100 GOSUB 29000
105 ? :? "LOADING 3D ATARI":? "by Harry
Morris  2/24/83"
110 RESTORE 200:READ J:DIM SHAPE1$(J)
120 FOR K=1 TO J:READ
P:SHAPE1$(K)=CHR$(P):NEXT
K:SHAPE1=ADR(SHAPE1$)
130 GRAPHICS 9:POKE 1175,0:FOR I=0 TO
15:POKE 1182,I:P=USR(SHAPTAB,10+I,
90+I,SHAPE1):NEXT I
140 GOTO 140
150 END
200 DATA 354,10,10,10,10,10,10,10,
10,10,10,10,10,10,26,42,42,42,10,106,
106,106,106,74,10,10,42,10,10,10,
10,10,10
202 DATA 10,10,10,42,10,26,74,42,74,42,
74,74,74,74,74,74,74,74,74,42,90,42,
90,42,74,74,74,74,74,74,74,74,74,74
204 DATA 74,74,74,51,51,51,35,35,35,3,
3,10,10,10,10,10,10,10,10,10,10,10,10,
10,10,10,10,10,10,10,10,10,10,10,10,10
206 DATA 10,10,10,10,106,106,106,106,
74,42,42,42,42,42,42,42,42,10,106,
106,106,106,74,74,74,74,74,74,
74,74,74,74
208 DATA 74,74,74,74,74,74,74,74,74,74,
74,74,74,74,74,74,74,74,67,67,67,35,
35,35,19,19,35,10,10,10,10,10,10,10,10
210 DATA 10,10,10,10,10,26,42,42,42,
10,106,106,106,106,74,10,10,42,10,10,
10,10,10,10,10,10,10,42,10,26,74,42,74
212 DATA 42,74,74,74,74,74,74,74,74,74,
74,42,90,42,90,42,74,74,74,74,74,74,
74,74,74,74,74,74,74,74,51,67,51,
19,3,10,10
214 DATA 10,10,10,10,10,10,10,10,10,10,
10,10,10,10,10,10,10,10,10,10,10,10,
10,10,10,10,10,10,42,42,74,106,
42,42,74,58
216 DATA 74,74,74,74,74,74,74,74,74,74,
90,74,106,74,74,74,58,74,42,74,74,74,
74,74,74,74,74,51,67,51,19,3,10,10,42
218 DATA 74,42,10,58,42,42,42,10,106,
106,106,10,10,10,10,10,10,10,10,10,10,
10,10,10,10,10,10,10,10,10,10,10,10,10
220 DATA 10,10,10,10,106,106,106,74,
42,42,42,42,42,42,10,106,106,255
999 REM ATARI3D  by Harry Morris  2/24/83
```

HOW TO USE SHAPEDIT

by
Craig Chamberlain

The SHAPEDIT utility was written to simplify the task of defining a shape table. This program makes it easy to draw and erase shapes in any graphics mode by using a joystick. It also has options to save and load shapes to any device, such as a cassette or disk. When you run the program, it takes a minute to initialize, and then prompts you with a bell and an asterisk displayed on the screen.

GRAPHICS MODE

Move the joystick left or right to choose a screen mode from three to eleven. This is just for editing purposes; the mode in which you edit your shape has no affect on the shape table produced.

PLAYFIELD

Move the joystick left or right to choose the current drawing color. The range is limited according to the current graphics mode, since some modes allow more playfields than others. The playfield number is one of the two data parts of each shape table instruction. This is similar to the COLOR statement in ATARI BASIC.

COLOR and LUMINANCE

These values are for your convenience in editing only, and are also selected by moving the joystick left or right. They are like the SETCOLOR statement. While editing, you can set the color and luminance for each playfield being used, but this information is not stored in the shape table. Your program which draws the shapes using SHAPTAB will have to include the necessary SETCOLOR statements.

CREATE

Press the joystick trigger once to start defining a shape. You will then be able to position the cursor anyplace on the screen. This defines the origin of the shape table being created. Once you have chosen a place to start the shape, press the trigger a second time. You will now be out of the positioning mode and in the drawing mode.

There are two ways to draw in this mode. To do point by point plotting, move the joystick in any of the eight possible directions. The cursor will react to your joystick selection (diagonal directions are somewhat difficult). Release the joystick and then press the trigger. This will cause the pixel to be officially plotted. You have now added an entry into the shape table. The entry includes the direction in which you just moved, and the playfield that was in effect at the time.

To do line drawing, keep the trigger pressed while moving the joystick. SHAPEDIT will not let you draw out of bounds. To draw out of bounds, you must return to the main menu, choose the CREATE mode again, and then change your starting position before you enter the drawing mode.

To exit the drawing mode and return to the main menu, press the OPTION key on the console.

The program remembers the previous starting position every time you enter the CREATE mode. It will reset this position only if you change to a graphics mode in which the previous position would be out of range.

There are three errors that may encountered while using the CREATE function of SHAPEDIT. The most common one says RANGE ERROR. This happens when you go from the positioning mode to the drawing mode, but the starting position does not allow part of the shape defined thus far to be drawn without going out of bounds for that graphics mode. This often occurs when you create a shape in one mode and try to display it in another mode of lower resolution. To recover from the error, first press the OPTION key to clear the error, then enter the CREATE mode again, this time using a different graphics mode or different starting position.

The second possible error message says NOT ENOUGH RAM, and also occurs during the transition from positioning to drawing modes. You will get this message if your computer does not have enough memory to support the requested graphics mode. This error will happen only when you try to access high

resolution modes (such as eight through eleven) on systems with less than full memory. Sorry, but there is no way to get around this error.

The last error can occur while you are in the drawing mode, and says MEMORY FULL. SHAPEDIT cannot hold more than 999 entries in the shape table, and will not let you CREATE shape tables longer than that limit. This should be no restriction, however, since it would be unusual to have a shape table that long. If it is necessary to change this limit, change the assignment to the variable EMAX in line 790 of the program.

If at any time you want to know how many entries are in the shape table, simply return to the main menu, where you will see the number of entries prominently displayed.

DELETE

This is similar to CREATE in that you press the trigger once to enter the position mode, then press it a second time to enter the erasing mode. When you enter the erasing mode, the current shape will be displayed in full. Every time you press the trigger, the last entry of the shape table will be removed and the shape will be redrawn. If you delete all of the entries, the program will automatically return to the main menu and will not let you enter the DELETE mode while the shape table is empty.

To delete the whole shape, all at once, press the SELECT key while in the main menu. This is called "clearing the shape."

SAVE

Use this option to store a shape to disk or cassette. Just as with CREATE and DELETE, press the trigger to choose this option. You will be prompted for the full device specification. This means D:filename for disk and C: for cassette. Our convention with disk systems has been to use the extender .SHP in the filename. To abort this command, press the RETURN key without entering anything. All I/O errors will be reported. It is not possible to select this option if your shape table is empty.

LOAD

Use this to retrieve a previously defined shape. This command is especially powerful because it functions like an append to the end of your current shape. If you already have a shape in memory, the shape being loaded will be added onto the end of your current shape. If you want to load a shape in place of the current one, you must first "clear the shape" using the DELETE mode discussed above.

The shape will not be loaded if it would cause the shape table to exceed the 999 entry limit, in which case an error message will be displayed.

That should pretty well cover all the details of the SHAPEDIT program. The program has been rather well "idiot proofed" and greatly facilitates the creation of shapes for use with SHAPTAB.

SHAPEDIT

by

Craig Chamberlain

```
100 GOTO 700
250 IF STRIG(0)=0 THEN 250
251 TRAP 275:P=USR(BELL,16):
ERR=3:GRAPHICS GM*(GM<9)+9*(GM>8)+16:
ERR=0:POKE COLR,(GM<9)+10*(GM>8)
252 POKE
708,10:HSET=HSET*(HSET<HMAX(GM)):
VSET=VSET*(VSET<VMAX(GM)):GOTO 257
253 JS=STICK(0):IF JS=15 THEN ON
STRIG(0)+1 GOTO 258,253
255 HSET=HSET+SX(JS):HSET=HSET*
(HSET<HMAX(GM))+HMAX(GM)*(HSET<0)
256 VSET=VSET+SY(JS):VSET=VSET*
(VSET<VMAX(GM))+VMAX(GM)*(VSET<0)
257 P=USR(SHPLOT,HSET,VSET):GOTO 253
258 GRAPHICS GM+16:POSITION
HSET,VSET:P=USR(BELL,16)
260 POKE 712,16*COL(0)+LUM(0):IF GM<9
THEN FOR K=0 TO 3:POKE
708+K,16*COL(K+1)+LUM(K+1):NEXT K
265 IF GM=10 THEN FOR K=0 TO 8:POKE
704+K,16*COL(K)+LUM(K):NEXT K
270 TRAP 310:RETURN
275 POP :GOTO 310
300 ERR=0
310 TRAP 310:GOSUB 500:ON NDX+1 GOTO
320,350,380,400,410,425,435,460
320 NDX=0:GOSUB 550:JS=STICK(0):IF JS=15
THEN 320
325 IF JS>12 THEN
PF=PF*(PF<=PFMAX(GM)):GOSUB 560:ON
JS-12 GOTO 350,460
330 GM=GM-3+SX(JS):GM=3+GM*(GM<9)+9*
(GM<0):P=USR(BELL,GM):GOSUB 560:GOTO 320
350 NDX=1:GOSUB 550:JS=STICK(0): IF JS>12
THEN ON JS-12 GOTO 380,320,350
360 PF=PF+SX(JS):PF=PF*(PF<=
PFMAX(GM))+(PFMAX(GM)+1)*(PF<0):
P=USR(BELL,PF):GOSUB 560:GOTO 350
380 NDX=2:GOSUB 550:JS=STICK(0): IF JS>12
THEN ON JS-12 GOTO 400,350,380
385 COL(PF)=COL(PF)+SX(JS):COL(PF)=
COL(PF)*(COL(PF)<16)+16*(COL(PF)<0):
P=USR(BELL,COL(PF)):GOSUB 560:GOTO 380
400 NDX=3:GOSUB 550:JS=STICK(0): IF JS>12
THEN ON JS-12 GOTO 410,380,400
405 LUM(PF)=LUM(PF)+2*SX(JS):LUM(PF)
=LUM(PF)*(LUM(PF)<15)+16*(LUM(PF)<0):
P=USR(BELL,LUM(PF)):GOSUB 560:GOTO 400
410 NDX=4:GOSUB 550:JS=STICK(0):IF JS=13
OR JS=14 THEN ON JS-12+2*(E=0) GOTO
425,400,460,400
411 IF STRIG(0) OR E=EMAX THEN 410
412 GOSUB 250:P=USR(SHAPTAB,SHAPE):IF
P>127 THEN ERR=P:GOTO 310
416 FOR K=0 TO 2:POKE
CURPOS+K,PEEK(90+K):POKE
84+K,PEEK(90+K):NEXT K
417 GET #6,P:POKE OLDAT,P:POKE COLR,PF
418 P=USR(SHPLUS):IF P=255 THEN 300
420 K=USR(BELL,P/16):POKE
77,0:E=E+1:S$(E)=CHR$(P):
S$(E+1)=CHR$(255):IF E<EMAX THEN 418
422 ERR=2:NDX=5:GOTO 310
425 NDX=5:GOSUB 550:JS=STICK(0):IF JS=13
OR JS=14 THEN ON JS-12 GOTO 435,410
426 IF PEEK(53279)=5 THEN
E=0:S$=CHR$(255):POSITION 9,4:? #6;"0
":GOTO 410
427 IF STRIG(0) OR E=0 THEN 425
428 GOSUB 250:P=USR(SHAPTAB,SHAPE):IF
P>127 THEN ERR=P:GOTO 310
429 IF STRIG(0)=0 THEN 429
430 IF STRIG(0) THEN ON (PEEK(53279)=3)+1
GOTO 430,300
431 P=USR(BELL,16):S$(E)=CHR$(255):
E=E-1:? #6;CHR$(125):POSITION HSET,VSET
432 P=USR(SHAPTAB,SHAPE):IF P>127 THEN
ERR=P:GOTO 310
433 IF E THEN 429
434 NDX=4:GOTO 300
435 NDX=6:GOSUB 550:JS=STICK(0):IF JS=13
OR JS=14 THEN ON JS-12 GOTO 460,425
438 IF STRIG(0) THEN 435
439 GOSUB 570:IF F$="" THEN ?
CHR$(125):GOTO 435
440 TRAP 490:OPEN
#1,8,AUX,F$:P=INT((E+1)/256):PUT
#1,E+1-256*P:PUT #1,P
450 P=USR(CIO,11,SHAPE,E+1):IF P>127 THEN
ERR=P:GOTO 310
455 CLOSE #1:? CHR$(125):OLDNDX=-1:GOTO
435
460 NDX=7:GOSUB 550:JS=STICK(0):IF JS=13
OR JS=14 THEN ON JS-12+2*(E=0) GOTO
320,435,320,410
462 IF STRIG(0) OR E=EMAX THEN 460
464 GOSUB 570:IF F$="" THEN ?
CHR$(125):GOTO 460
466 TRAP 490:OPEN #1,4,AUX,F$:GET
#1,P:GET #1,K:K=K*256+P-1
470 IF E+K>EMAX THEN ERR=3:CLOSE
#1:GOTO 310
475 P=USR(CIO,7,SHAPE+E,K+1):IF P>127
THEN ERR=P:GOTO 310
485 CLOSE #1:? CHR$(125):E=E+K:POSITION
9,4:? #6;E;"    ":OLDNDX=-1:GOTO 460
```

MICRO CHIP INC.

HOME COMPUTERS & SOFTWARE

Open 7 Days A Week
10-9 Mon.-Sat. 12-5 Sun.

HARDWARE

The FRANKLIN ACE 1000 —— the Apple II+ compatable computer at an affordable price.

The INCREDIBLE
MORROW DESIGNS
MICRODECISION 17 —— 64k and double density disc drive for only $1550. — plus $2000. worth of software **FREE.**

Including: Wordstar-Correctit, Logic Calc., CP/M 2.2, Microsoft Basic and Northstar Bazic.

Order double disc drive version for $1950. and get Pearl Data Base Management software **FREE.**

ORANGE PEEL —— another Apple compatable computer with a choice of two detachable keyboards; 57 keys or 100 keys (includes 10 microprogrammable keys).

ATARI 400 and 800

SOFTWARE

We support Apple, Atari and IBM with over 800 software titles.

MICRO CHIP

3628 Rochester Road, Troy, Michigan
Century Plaza Shopping Center
Phone: (313) 524-1230

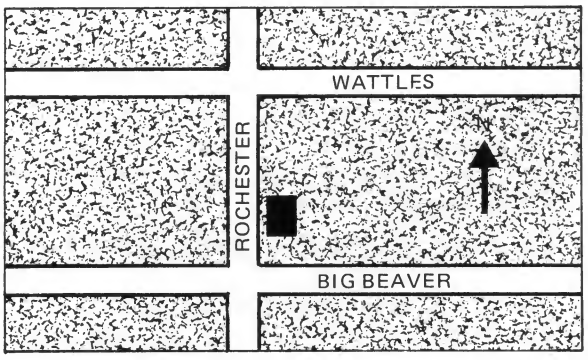
SOFTWARE RENTAL

Try the latest programs before you buy — for as little as $2.00 per day (MicroChip Club members only).

Join the club and get low rentals, plus member discounts on hardware and software.

COMPUTER CLASSES

- Basic — from beginning to advanced
- Assembly Language
- Word-Processing
- Electronic Spred Sheet and other business applications



Wabash Diskettes —— only $19.95

We also handle a broad line of printers and other peripheral devices.

```
490 ERR=PEEK(195):CLOSE #1:GOTO 310
499 END
500 GRAPHICS 1:POKE 77,0:POKE
708,0:DL=PEEK(560)+256*PEEK(561)
505 POKE DL+6,7:POKE DL+9,48:POKE
DL+11,48:POKE DL+20,112:POKE
DL+21,7:POKE DL+22,48:POKE DL+23,48
510 ? #6;? #6;" SHAPTAB EDITOR":? #6;"
BY":? #6;" CRAIG CHAMBERLAIN":? #6;"
ENTRIES:";E
520 ? #6;" GRAPHICS MODE  ";GM
521 ? #6;"    PLAYFIELD  ";PF
522 ? #6;"       COLOR  ";COL(PF)
523 ? #6;"    LUMINANCE  ";LUM(PF)
524 ? #6;"      CREATE"
525 ? #6;"      DELETE"
526 ? #6;"       SAVE"
527 ? #6;"       LOAD"
530 POKE 708,10:IF ERR=0 THEN 540
531 IF ERR=141 THEN ? #6;" RANGE
ERROR":GOTO 535
532 IF ERR=2 THEN ? #6;" MEMORY FULL"
533 IF ERR=3 THEN ? #6;" NOT ENOUGH RAM"
534 IF ERR>3 THEN ? #6;" I/O ERROR ";ERR
535 ERR=0:IF PEEK(53279)<>3 THEN 535
540 POSITION 0,13:? #6;"
":OLDNDX=-1:RETURN
550 IF NDX=OLDNDX THEN RETURN
551 P=USR(BELL,16):POSITION
15,5+OLDNDX:PUT
#6,0:OLDNDX=NDX:POSITION 15,5+NDX:PUT
#6,42
552 IF STICK(0)<>15 THEN 552
553 POKE 77,0:RETURN
560 POSITION 17,5:? #6;GM;" ":POSITION
17,6:? #6;PF;" "
562 POSITION 17,7:? #6;COL(PF);"
":POSITION 17,8:? #6;LUM(PF);" "
565 IF STICK(0)<>15 THEN 565
569 RETURN
570 P=USR(BELL,16):? CHR$(125):POKE
752,0:? "PLEASE ENTER DEVICE
SPECIFICATION"
575 POKE 764,255:INPUT F$:
AUX=0
580 POKE 752,1:? " ";:RETURN
700 DIM COL(15),LUM(15),PFMAX(11),
HMAX(11),VMAX(11),SX(15),SY(15),
M$(421),F$(15)
705 FOR K=0 TO 15:COL(K)=0:LUM(K)=0:NEXT
K:GM=3:GOSUB 500
710 FOR K=0 TO 11:READ
P,X,Y:PFMAX(K)=P:HMAX(K)=X:
VMAX(K)=Y:NEXT K
715 FOR K=0 TO 14:READ
X,Y:SX(K)=X:SY(K)=Y:NEXT K
720 M=ADR(M$):CIO=M:SHAPTAB=M+34:
VBLSET=M+105:BELL=M+115:SHPLOT=
M+148:SHPLUS=M+186
725 COLR=1641:OLDAT=1642:CURPOS=1643
730 FOR K=1 TO 34:READ
P:M$(K)=CHR$(P):NEXT K
740 FOR K=1536 TO 1640:READ P:POKE
K,P:NEXT K
750 FOR K=1 TO 387:READ
P:M$(34+K)=CHR$(P):NEXT K
760 P=USR(VBLSET,1536)
790 E=0:EMAX=999:DIM S$(EMAX+1):S$=
CHR$(255):SHAPE=ADR(S$):TRAP 310:GOTO
320
800 DATA 3,40,24,4,20,24,4,20,12,3,40,
24,1,80,48,3,80,48,1,160,96,3,160,96,
3,320,192,15,80,192,8,80,192
805 DATA 15,80,192
810 DATA 0,0,0,0,0,0,0,0,0,0,1,1,1,-1,
1,0,0,0,-1,1,-1,-1,-1,0,0,0,0,1,0,-1
850 DATA 104,162,16,104,104,157,66,3,
104,157,69,3,104,157,68,3,104,157,73,3,
104,157,72,3,32,86,228,132,212,169
852 DATA 0,133,213,96,173,104,6,170,41,
15,240,7,202,142,104,6,142,1,210,76,
98,228,173,23,228,72,173,22,228,72,152
854 DATA 96,173,21,228,72,173,20,228,
72,96,255,255,0,1,1,1,0,255,255,255,
243,217,193,182,162,144,128,121,108,96
856 DATA 91,81,72,64,60,53,47,22,22,10,
22,46,46,94,94,190,190,190,190,38,18,
18,38,78,78,158,158,62,78,78,78,255
858 DATA 2,6,255,4,3,5,255,0,1,7,255,
255,255,255,255,160,104,104,133,209,
104,133,208,160,0,177,208,48,45,230,208
860 DATA 208,2,230,209,72,74,74,74,74,
170,189,37,6,24,101,84,133,84,189,
39,6,170,24,101,85,133,85,138,48,1,152
862 DATA 101,86,133,86,104,41,15,168,
32,18,6,152,48,85,162,2,181,90,149,84,
202,16,249,208,192,104,104,170,104,168
864 DATA 169,7,76,92,228,104,169,3,
141,50,2,141,15,210,173,104,6,41,240,
141,104,6,9,12,168,104,104,170,189,47,6
866 DATA 141,0,210,140,104,6,96,104,
162,2,181,90,149,84,202,16,249,160,0,
32,18,6,104,133,86,104,133,85,104,104
868 DATA 133,84,172,105,6,32,18,6,
132,212,169,0,133,213,96,104,169,
255,141,110,6,169,8,141,31,208,173,
31,208,41
870 DATA 4,208,8,162,255,134,212,232,
134,213,96,173,120,2,73,15,240,56,141,
111,6,169,0,164,87,78,111,6,174,107
872 DATA 6,208,1,24,42,78,111,6,144,6,
190,64,6,236,107,6,42,78,111,6,174,
```

```
108,6,208,6,174,109,6,208,1,24,42,
78,111
874 DATA 6,144,22,192,8,208,12,56,176,
4,240,102,208,169,174,109,6,240,6,
190,76,6,236,108,6,42,170,189,88,
6,48,82
876 DATA 205,110,6,240,229,141,110,6,
162,2,181,90,149,84,202,16,249,172,
106,6,32,18,6,174,110,6,189,37,6,24,109
878 DATA 107,6,133,84,133,90,189,39,6,
170,24,109,108,6,133,85,133,91,138,48,
2,169,0,109,109,6,133,86,133,92,32
880 DATA 28,6,141,106,6,162,2,181,
90,149,84,202,16,249,172,105,6,32,18,
6,173,132,2,208,149,173,110,6,48,
144,162
882 DATA 2,181,90,157,107,6,202,16,248,
173,105,6,141,106,6,173,110,6,10,10,
10,10,13,105,6,133,212,232,134,213,96
999 REM SHAPEDIT by Craig Chamberlain
2/24/83 COPYRIGHT 1983 ALLIANCE
SOFTWARE
```

Review of Larry Atkin's Chess 7.0

by Ted C. Lambert

The human race has taken another step toward oblivion. A program for a microcomputer now exists that plays better chess than I do. Listen, I know about things like the Opposition, pinning, multiple attacks, masked attacks, illusory guard, the power of a rook on the seventh rank, and all kinds of stuff about passed pawns. So does this program. At level 1, I know, I tested it. At level 2 and above (except M), it takes advantage of its library of openings, stored on disk. It's a putzer only at level 0, which takes 1 second per move.

Chess 7.0 has a total of 17 levels to choose from. Levels 3-8 are limited by increasing amounts of time. Levels A-F are limited by increasing depths of lookahead. That is, level B will look ahead at least two moves, however long that takes, and level 4 will "think" for no more than 80 seconds. Levels 0-2 are "introductory levels," which means they play real fast, and aren't as strong as higher levels, though I hate to think how many years it took me to learn to play that well.

There are also two special levels, P and M. Level M specializes in finding checkmates. Set up a position, pick level M, and if there's a way it can checkmate you, it will find it. You can use this for solving chess problems. P is for perpetual. Unless stopped, this level will look up to 23 ply (or half-moves (= 12.5 moves)) ahead. In most cases, you would need extreme patience to wait that long. In fact, your unborn grandchildren might have difficulty waiting that long. The User's Guide recommends this option for postal chess.

The User's Guide, written by David Harmon, is very good. It's glossy, thick, and logically organized. It's 64 pages long, has a Table of Contents, an Index, some appendices with interesting general articles on chess, step-by-step instructions on how to use the program, and 18 pages describing the available features. This program has features!

If you request it, this program will: run a demo to show you how its features work, play either side of the board, take both sides and play against itself, take neither side so you can play your own self, record the moves, give you a numerical evaluation of its position (positive is good for it, negative good for you), tell you what move it's actively considering, give you advice (recommends your best move), let you switch sides at any point, let you take back moves, tell you which moves are legal for a selected piece, tell you which pieces attack and protect a selected square, save games to disk, retrieve games from disk (including some instructive ones that come with the package), and even allow a "camouflaged" chess, in which you have to memorize who moved what to where. That's as close to blindfold chess as it could come.

To use Chess 7.0, you need a keyboard or paddle controller (joysticks won't work), disk drive, and, of course, the cpu and TV. You must keep the disk in the drive throughout the game, since the program refers to the disk for special feature programming and openings. That library of openings gives it a powerful headstart over other micro chess programs. Neither humans nor computers can derive decent openings from scratch. This program also recognizes more drawn positions than some others. It declares a draw due to repetition of moves, or when neither side has enough material to mate. It even knows the 50 move rule. Another attractive capability is it "thinks" on your time. That is, while you're plotting your move, it's already examining possible responses.

It also predicts your move. If you make the expected move, it may reply much quicker than normal. I hadn't thought a chess program could take advantage of human psychology, but this does, and I think Larry Atkin designed it this way on purpose. If the program replies instantly, you think["Oh no, I'm doing just what it wants." It uses intimidation. Asking its advice has the same effect. I feel reluctant to make a move my opponent recommends, even though it may be the best move. I always want to demonstrate the superiority of human intelligence over machine intelligence. Pride, foolish pride. That cost me a few games.

It plays fast, too. At level 4, it can beat Atari Chess at any level. But level 4 takes only 40-80 seconds to move. You can play a

tough game of chess within an hour. That I like.

This program does have some drawbacks: 1) it costs $70, 2) you must have a disk drive, 3) the list of options on the bottom and right side of the screen makes an otherwise good display look busy (the board display is basic black and white, and looks similar to the diagrams found in chess books), and 4) it's really too strong for an inexperienced player. If you think the Sicilian Defense is something Marlon Brando used in "The Godfather," then this game is not for you.

If you want an opponent that will make you play better, look no further. To those of us who feel this way, the real value of a chess program is how well it plays.

This program is MEAN.

I set up some classic endgame positions, all designed to test its knowledge of chess principles, rather than the ability to look ahead a certain number of moves. Computers are notoriously bad at endgames because in endgame positions you would often have to look ahead more than 15 moves (by each side), to see how to queen a pawn. This is approximately 10 to the 30th possible combinations, in a SIMPLE position. A Cray 2 would choke on that, let alone an Atari 800. But the moves are easy, if you know the principles involved. The real brains of a chess program are the routines that evaluate its position based on vague and sometimes conflicting chess-playing principles. Explaining these principles to a computer is the hardest part of writing a chess program.

Larry Atkin has a lot of experience, though. He was one of the three people who originally wrote the famous Northwestern University chess program, the one that in its various versions has pretty well dominated the annual World Computer Chess Tournament. So it's understandable that his mental offspring passed all my endgame tests at its second lowest level, even when Atari Chess failed most of them at its highest level. What surprised me was losing at levels 3, 4, and 5 when I thought I had entered the endgame with an advantage. I can tolerate it being my equal at the endgame, but being my better bruises my ego.

In short, it's good. I like it. So much so, I'm gonna buy this copy that the Family Computer Center loaned us for review. If you get it (it's marketed by Odesta), good luck.

If you want to know more about chess, I strongly recommend the following two books: "The Game of Chess" by Dr. Siegbert Tarrasch, and "My System" by Aron Nimzovich, both published by Tartan books. There are many books on computer chess, including "Chess and Computers" by David Levy, who made the famous 10-year bet that no computer program could beat him in a match by August, 1978. He won, against the Northwestern program.

Assembler SIG

By Phil Heavin, Secretary

February Meeting Minutes

This months meeting was something unexpected for me, it was even better than January's meeting! Tom came up with an even better method of animating objects in assembly language. This fairly simple method uses only integer addition to achieve many types of motions used in video games such as, bullets, rocket ships, billiard ball roll and bounce, sinusoidal and circular motion. There is sure to be more about this in future sessions of SIGASM.

Next I gave a demonstration of Dunion's Debugging Tool (DDT), available from APX (APX-20150 $39.95). This incredible tool is a must for any ATARI Macro Assembler owner. This program did more than live up to my expectations. The fact that I was able to demonstrate it with only a few hours of experience speaks well for the quality of this product.

April's Meeting

April's meeting will be Thursday, the 7th at the home of Pat McCabe in Mount Clemens. You can contact Pat at (313)-791-0946 or me at (313)-939-6213. The meeting will begin at 7:00 with socializing and free form discussion with the actual business portion starting at 7:30. We hope to see you there.

MAC/65

by Sheldon Leemon

Until fairly recently, the assembly-language programmer did not have much to work with in the way of tools for developing programs on the Atari. Atari's Assembler-Editor cartridge was so limited in its capabilities and so slow that when the Atari Macro-Assembler (AMAC) came along, it was hailed as a major breakthrough. Now, Optimized Systems Software, which produced Atari's cartridge, has come up with a new macro-assembler, MAC/65. Though the features of this assembler are roughly simlar to those of AMAC, the design philosophy and overall implementation of MAC/65 are so far superior that for most people it is clearly the better choice.

The most obvious difference between AMAC and MAC/65 is the way in which the user prepares source code. AMAC provides a separate text editing program, MEDIT, on the program disk. It is a very powerful editor with complete cursor control, block movement, insertion and deletions, and many other features associated with the best word-processing programs. But for the preparation of routine source code, such features are not really necessary. There are so many different commands that it could take longer to learn how to use the editor than AMAC itself. On top of this, the editor is not co-resident with the assembler, and must be loaded in separately each time it is used. Since Atari, in its infinite wisdom has seen fit to copy-protect the AMAC disk, you cannot keep the assembler, editor and source code all on the same disk. This means that if you are using only one disk drive, every time you want to make even the slightest change to a source file, you have to boot DOS, load the editor, swap disks, load the source file, change the source code, save the source file to disk, swap disks, boot DOS, load the assembler, and then swap disk so that the assembler can then read the source file back in. The poor new user who is a sloppy typist, or who is prone to syntax errors will go through this painstaking process time and again before getting an error-free assembly. Given the maddeningly slow access time of Atari disk drives, it soon becomes apparant that the only reasonable way to work with this system is to use a Ramdisk. But the copy protection prevents you from making the DOS modifications necessary to use the Ramdisk without jeopardizing your copy of AMAC. This puts the user in the absurd position of having to break the protection scheme on a piece of software he has paid for, in order to make proper use of it.

Contrast this with the approach that OSS takes. MAC/65 has a co-resident line editor, that is an improved version of the editor used on the cartridge (which is itself an enhanced version of the Atari BASIC line editor with which every user is familiar). It is extremely simple to use, and adequate for most routine editing. In this version, it even does some simple syntax checking at entry time! Because MAC/65 builds on the foundation of the Assembler cartridge, it is upwardly compatible with it, and source files created with the cartridge can be used with MAC/65 without change. With MAC/65, you can now save these source files in a tokenized format. This not only speeds up the loading of these files, but substanitally reduces the size of the file on disk, so that even with the addition of line numbers, MAC/65 source files are generally no larger than equivalent unnumbered AMAC ASCII files. MAC/65 is unprotected, so you can keep the assembler on the same disk as a rather large source file, and avoid swapping disks if you only have one drive. In contrast to the AMAC editing session described above, you could boot DOS and automatically run MAC/65, load the source file, assemble to a free area of memory, re-edit the file which is still in memory, reassemble, run the program and test it, go back to the MAC/65 editor and make changes, all without having to reload DOS, source code, editor, or assembler.

Perhaps best of all is the fact that MAC/65 gives you a choice of the type of editing you may use. If you prefer to use your favorite text editor to prepare source files without line numbers, you may do so. MAC/65 allows you to enter such files, and will automatically append line numbers. Since the PRINT command allows you to save numbered source files in an unnumbered format on disk, you can switch between the two systems quite freely. This is a perfect solution to the problem of trying to add a substantial amount of code to the middle of a program without adjusting the line numbers.

As far as the features of the actual assembler are concerned, MAC/65 is almost identical to AMAC. Both assemblers are extremely fast. The one large source file I have, about 1000 lines long, took two minutes to assemble on AMAC and only one minute on MAC/65. This appears to be due mainly to the speed at which source files are read in; when I

loaded the entire source file into memory using MAC/65 the assembler took only four seconds to produce the 1.5K of object code! This ability to assemble from memory as well as from disk files without having to read the source in each time appears to give the speed advantage to MAC/65.

The types of operators that these two assemblers recognize are generally the same, although each accepts a few not found on the other, such as Shift Left, Shift Right, Modulus, and Length of String on AMAC, and .REF, and logical as well as bitwise AND, OR, XOR, and NOT on MAC/65. The pseudo-ops are likewise mostly the same, with a few significant exceptions. MAC/65 has better text handling capabilities. A .SBYTE op assembles text in the internal screen code format, while the .BYTE op allows longer strings than AMAC, and allows you to add a numeric offset to each character, such as 128 which would set the top bit and generate inverse characters. On the other hand, it lacks AMAC's REAL16 pseudo-op, which converts numbers to the six byte format utilized by the floating-point routines, as well as some of the flexibility in the kinds of information that can be generated in the assembly listing. It would be particularly helpful if MAC/65 could show where labels were referenced, as AMAC can. Also, .INCLUDE statements cannot be nested in MAC/65.

MAC/65 has full macro capabilities, a little fuller perhaps than AMAC. For one thing, MAC/65 allows the passing of string as well as numeric parameters. If the first parameter you pass is "Hello", the macro expression %$1 is expanded to those ASCII characters, and the expression %1 returns the length of that string expression. MAC/65 allows the passing of up to 63 parameters per macro call. The expression %0 returns the total number of parameters passed in a specific call. The only expression it lacks relative to AMAC is the %K, which returns a unique serial number for each macro call.

Probably the biggest advantage that MAC/65 has over AMAC in regard to macro capabilities, however, is in its documentation of those capabilities. The AMAC manual devotes only a page or two to a brief and fairly technical description of macro expressions. There are no real explanations or examples of typical applications. MAC/65, however, devotes 13 pages of its manual to a full explaination of the macro capabilites. The explanation is written in a straightforward, nontechnical style, with illustrative and useful examples on every page. In addition, a whole library of macro I/O

routines are included on the program disk. This library includes such macros as PRINT, OPEN, and INPUT, which can be used almost exactly like their BASIC counterparts. Statements like:

```
PRINT    "hello, world"
           and
OPEN     1,4,0,"P:"
```

will expand out to the whole ten or fifteen line program needed to accomplish these tasks in assembly language. With this kind of help, assembly language programming could almost be fun! Along with the macro library, the disk contains a sample program which gives a practical demonstration of the use of the library macros.

Although this is a review of MAC/65, some mention should be made of the fact that the program disk contains two additional, and rather substantial, systems software packages. One is OS/A+, an improved DOS with many convenient features, such as instant access to resident utilities. This program formerly sold for $75 by itself, but is now included free with MAC/65. The other is BUG/65, a fully relocatable 8K interactive debugging moniter. Since this debugger has 22 different commands, it would be impossible to adequately cover it here. Suffice it to say, it is a powerful and much-needed debugging tool that nicely complements the assembler.

But perhaps the best reason to recommend the purchase of MAC/65 is the dedication which OSS has shown to serving its customers. Their record for delivering a high level of support sets them apart from larger software vendors like Atari, who tend to abandon a customer after the sale. Their system software is not copy protected, so that the user can enjoy its full power and convenience. When bugs turn up in OSS products, they usually fix them quite promptly, and supply the fix to all registered owners. They are responsive to customer complaints and suggestions. They constantly work at improving their product line, and provide these improvements at a nominal cost to their customers. And they design their programs with an eye to the future, so that newer versions will be upwardly compatible with current software. Such an attitude can be as important to the customer as the software itself, and in the case of MAC/65 it makes an already fine product an almost irresistable bargain.

THE ATARI BASIC COMPILER

DATASOFT INC(R)

A REVIEW
BY DAVID T. FROMME

I Have spent the past year developing an elaborate multi-tier distributor accounting and inventory system. While the system is good it suffers one serious malady- slooowwnesss. So when I became aware of the pending release of a compiler for ATARI basic by Datasoft I rushed to get my order in. After several delays I finally got my copy towards the end of January and set right in to see what it could do to alleviate my problem. After several weeks working with it I feel that I have some comments worth sharing.

First the good news: it does make programs execute considerably faster! The speed enhancement varies depending on the type of program and whether you elect to use the floating point (slower) or integer (faster) option. Execution time is enhanced by a factor ranging from 3 to 20 times over the normal interpreter. In addition the compiler is easy to use and is reasonably well documented (about an 8 on a scale of 10). The prompts and various control messages are excellent. Time for a compilation is a little slow to my liking but is not a major drawback. Of course if you have a bug you have to go back to the interpreter to change the program and then start the compiler up again. This necessitates a lot of rebooting and putting the cartridge in and out, but is the nature of compilers in general and not a fault of Datasoft's compiler.

The compiler works with the tokenized basic modules that you create by doing a SAVE from the interpreter (Basic cartridge) so there is no need to use the LIST option instead. It takes the tokenized input and creates several intermediate assembler segments that are then brought together in a final pass into an executable object module that is approximately 20% larger (plus 30 sectors) than the original Basic module. The extra 30 sectors are the run time routines that replace many of the modules that were in your Basic cartridge.

The resulting object module brings up the first questionable area about the compiler. For most users the nature of the object module will be a definite plus: it can be executed via a BINARY LOAD from DOS or renamed as AUTORUN.SYS and be used a self loading module. This is great for games and single purpose programs. It is a serious detrement when you have a series of programs that are part of an integrated system. The RUN statement cannot be executed from within a program (a fellow worker is busy trying to break this barrier and hopefully we will be able to pass the key on to you at a latter date).

The feature that has caused me the most difficulty is restrictions that the compiler places on the use of CHARACTER data and SUBSTRINGs. First all your CHARACTER DIMs must be increased by one byte, Second the use of substrigs to the left of an equal sign (ALPHA$(1,10) = "XXXXXXXXXX") is not practical and must be replaced by the use of PEEKS and POKES. While the method is not difficult it doesnt work with the interpreter in all cases. This means that you cannot test out your program with the interpreter when you introduce these changes. This incompatability is in my opinion the compilers worst drawback.

Other no-nos are:

-No variable dimensioning.
-The use of variables for GOSUBS and GOTOS is not permitted (this is very diheartening after spending hours trying to save memory)
-DOS and other similar commands are not allowed.

The size of the program that can be compiled is limited by the amount of disk space availabe and will normally be in the vicinity of 100 sectors for the original BASIC program. I find this restriction very frustrating since I have dual density drives but cannot get the compiler to execute in dual density mode because of the manner in which it has been copy protected. The restriction exists becuase the intermediate assembler modules must be written to drive one Which must be in single density mode to accept the compiler. If the assember segments could be directed to drive two I could double the size of the programs which could be compiled.

I have been in touch with Datasoft about this problem and several bugs I encountered. So far the known bugs are as follows:

-numeric subscripts cause the compiler to bombout.
-The VAL function does not work.
-Successive operators (NUM=NUM*-1) dont work and have to replaced(NUM=0-NUM)
-TRAP 32767 has to be replaced with TRAP 40000.

Datasoft will hopefully fix these problems and I am sure that it will eventually meet the high standards of their other software. While I have met some frustrations I feel that the benefits of the compiler are well worth it.

[Editor's note: I have been informed that the first release of the Datasoft compiler is incompatible with the Newell Fastchip, a popular replacement for the standard Atari floating-point ROM. I have also been advised of another Basic compiler recently released by Monarch Enterprises. It reportedly avoids most of the pitfalls of the Datasoft compiler. It handles substrings in the same manner as ATARI Basic and is claimed to reduce your progam's size by 20 to 50 pecent. It's one big drawback is that it doesnt offer a floating point option. It will handle integer numbers up to +-80,000,000 though. Hopefully someone will be able to submit a review of the Monarch compiler in the near future.)

MUSICBOX

by Jerry White

Reviewed by Greg Harrison

This package lets you convert Atari's MUSIC COMPOSER files into a VBI (vertical blank interrupt) routine. This lets you play the music while a basic program runs unaffected by the music.

MUSICBOX is made up of the following six programs:

MENU- Lets you access all the programs on the disk with one key input.

CONVERT- This program converts the MUSIC COMPOSER files to the MUSICBOX version.

PLAYTEST- This one reads the converted files and plays them. While the music plays, it gives some options. You can list the program, shut off and restart the music, or return to the menu. You also can delete some of the lines in this one to accomodate your own program.

COLORGRAN- This program also reads the converted files, but displays a graphical scene with the colors flashing to the beat of the music.

MUSICBOX- This displays the full range of the Atari MUSIC COMPOSER cartridge. With the joysticks or keyboard, you can move a marker about the musical staff. As you move it, it plays the note, and shows the MUSIC COMPOSER and SOUND command equivelant.

TRANSLATE- Originally from the magazine A.N.A.L.O.G., this program reads the MUSIC COMPOSER files and optionally prints or displays the commands typed to create the music file on MUSIC COMPOSER.

Musicbox also comes with several sample songs in both MUSIC COMPOSER and MUSICBOX format. The instruction booklet is very well done. It includes detailed descriptions of each program as well as a complete listing of each. At the end of the booklet it gives a list of musical chords. The given numbers allow you to put the chords into a BASIC program via the SOUND command.

There's just one problem. The 'CONVERT' program doesn't recognize ties or slurs. Every note is separated. But it doesn't really affect the sound of the song since it's barely noticeable.

Overall, the MUSICBOX package is a very nicely done piece of software and a very usefull utility to have. My personal rating is an A-.

MACE UNCLASSIFIEDS

FOR SALE: Pre-owned software. Preppie, Apple Panic, Bug Attack, Pool 1.5, My First Alphabet, Rear Guard: $15 each, all original with documentation. STEREO EQUIPMENT: JVC KD-75 cassette deck - $149 ($380 new). Pioneer RT-707 reel to reel deck - $349 ($695 new). Call Fred - (313)-375-1225.

(MEMORY CHIPS FOR SALE - All working RAMS - pulled from Atari memory boards: 24 - 4116s: $1 each; 16 - 4108s (6 of them test as good as 4116s), 50 cents each. Also, 2 2708 EPROMS (unprogrammed): $2 each. 30 bucks takes the whole lot. Claus Bucholz, 2216 Leif Ave., Muskegon MI 49441. (616)-755-3396

WANTED: Used educational software - disk or cassette. Especially interested in Spinnaker brand programs. Call G. Baker - (313)-355-3892.

TRADE: BASIC cartridge for 16k RAM board or best offer. Call George Friederichs (313)-329-7429.

WHAT A DEAL! Atari 2600 VCS with 30 cartridges/games, keyboard controllers, new joysticks - $500. Call Mike (313)-978-8432

FOR SALE: Microtek 32k RAM board. Works in 400 or 800. $85.00 or will trade. Pete Vellenga, 2742 Abigail S.E. Grand Rapids, MI. 49506. (H) (616)-949-1603, (W) (616)-942-9300.

LIGHTPEN DISCOUNT FOR MACE MEMBERS

Order a Symtec Light Pen (see ad on page 11) direct from the manufacturer before March 31,1983, and receive a $50 MACE member discount off of the retail price of $150! Don't confuse this pen with cheap imitations! The Symtec light pen is a rugged, industrial quality unit with excellent accuracy.

MICHIGAN ATARI COMPUTER ENTHUSIASTS

P.O. Box 2785
Southfield, MI 48037

24HR BULLETIN BOARDS 589-0996/274-3940

MACE HOTLINE (Meeting Info) 338-6837

PRESIDENT
Marshal Dubin
2639 Hempstead
Auburn Heights, MI 48057
338-6837

VICE-PRESIDENT
Jerry Aamodt
4148 Huhn
Rochester, MI 48063
574-1020

TREASURER
James Phillips
40008 Cambridge
Bldg 23 - Apt 103
Canton Township, MI 48187
981-1523

CORRESPONDING SECRETARY
Mike Lechkun
32229 Ruehle
Warren, MI 48093
978-8432

RECORDING SECRETARY
William Black
1251 Duckwood
Milford, MI 48042
887-6870

PROGRAM COORDINATOR
Gretchen Levitan
12709 Borgman
Huntington Woods, MI 48070
399-6964

DISK LIBRARIAN
Chet Gonterman
35088 Savannah Lane
Farmington Hills, MI 48018
553-7443

CASSETTE LIBRARIAN
Faris Ajo
6642 Leytonstone
West Bloomfield, MI 48033
661-0388

NEWSLETTER EDITOR
Arlan R. Levitan
12709 Borgman
Huntington Woods, MI 48070
Compuserve: 70675,463
Source ID: TCT987
Voice: 399-6964

NEXT MEETING 04/19/83

Southfield Pavillion
Ten & a Half & Evergreen
7:00 PM

M.A.C.E.
P.O Box 2785
Southfield, MI 48037

$20.00 fee for 12 months

M.A.C.E. MEMBERSHIP APPLICATION

Name _____ Phone _____
Street _____
City _____ State _____ Zip Code _____
Company (if applicable) _____
System Description _____ Disk/Tape _____
Suggestions _____
I can help with . . . _____

[ ] NEW [ ] RENEWAL

If Renewal, MACE #_____

*Make checks payable to:
M.A.C.E.

Date _____ Coupon _____
Amount _____ Membership Card Number _____
☐ Cash Expires _____
☐ Check Number

PROTECT YOUR INVESTMENT WITH Stitcher™ QUALITY.

- DOUBLE-STITCHED SEAMS
- CUT-OUTS FOR WIRING
- NYLON THREAD
- SOFT, STRONG, 27 OZ. VINYL
- CUSTOM STITCHER FIT
- 100% WORKMANSHIP WARRANTY
- THOUSANDS SOLD WITHOUT A RETURN

ATARI*	
400	14.89
800	18.89
410	7.89
810	11.89
830	6.89
820	13.89
825	17.89
IBM-PC*	
Keyboard	15.89
Monitor	21.89
Memory Unit	25.89
Mon. & Mem.	39.89
T.I.*	
99/4A	14.89
Monitor	25.89
COMMODORE*	
64	13.89
20	13.89
Datasette	7.89
Disk Drive	11.89
EPSON*	
MX70-MX80	17.89
MX80 FT	18.89
MX100	24.89
APPLE*	
II	18.89
II Keyboard	12.89
II Stack	39.89
(without monitor stand)	
II Stack	39.89
(with monitor stand)	

APPLE*	
III	19.89
III Keyboard	13.89
III Stack	39.89
Disk Drive	8.89
Stacked Pair	12.89
Monitor	25.89
FRANKLIN*	
1000	18.89
Keyboard	12.89
OKIDATA*	
82A	17.89
83A	24.89
PERCOM*	
Master	9.89
Slave	8.89
Pair	11.89
Dual	11.89
ZENITH*	
12" Green	25.89
SANYO*	
9" Green	18.89

COLORS
WHEAT
BROWN
CHARCOAL
BLACK

*Registered trademarks of Apple Computer Inc., Atari Inc., Commodore Electronics Ltd., Espon of America Inc., Franklin Computer Corp., International Business Machines Inc., OKI Electric Industry Company, Percom Data Corp., Sanyo Electric Ltd., Texas Instruments Inc., Zenith Data Systems.

NOW AVAILABLE IN OVER 50 STORES NATIONWIDE

Dealers Call Collect: 1-313-979-1698

Stitcher™ Inc.

P.O. Box 68, Sterling Heights, MI. 48078

Name_____

Address_____

City_____

State_____ Zip_____

Payment Method
☐ Check
☐ Credit Card

Master_____ Visa_____ Exp._____

MFR/MODEL	QTY.	COLOR	PRICE EA.	PRICE TOTAL

(In Michigan add 4% Tax.) Postage and Handling $1 50

TOTAL

MICHIGAN ATARI COMPUTER ENTHUSIASTS
P.O. BOX 2785
SOUTHFIELD, MICHIGAN 48037

BULK RATE
U.S. POSTAGE
PAID
PERMIT #431
SOUTHFIELD, MI

01/84

IMPORTANT DATED MATERIAL

PLEASE DO NOT DELAY

Printing and Bindery Services by • GRAPHIC ENTERPRISES, INC. Detroit, Michigan • 313-839-6800